Predictive Maintenance by Detection of Gradual Faults in an IoT-Enabled Public Bus

1st Gautam Vira
Faculty of Engineering
University of Ottawa
Ottawa, Canada
gvira022@uottawa.ca

2nd Patrick Killeen
Faculty of Engineering
University of Ottawa
Ottawa, Canada
pkill013@uottawa.ca

3rd Tet Yeap
Faculty of Engineering
University of Ottawa
Ottawa, Canada
tyeap@uottawa.ca

4th Iluju Kiringa
Faculty of Engineering
University of Ottawa
Ottawa, Canada
iluju.kiringa@uottawa.ca

***Abstract*—The widespread incorporation of Internet of Things (IoT) devices in various systems such as mobile phones, vehicles, and security systems is used to collect data. These large volumes of data can be used to train models to make predictions about various things. One such application that uses data from IoT devices is called predictive maintenance. Predictive maintenance involves collecting data from machines and using algorithms to analyze the machine's condition or determine if the machine requires maintenance or repairs. The work in this paper presents an algorithm to perform predictive maintenance by detecting potential faults, and the algorithm is tested for the cooling system of a bus. We also generate a synthetic time series dataset to simulate normal buses, and buses with underlying intermittent faults and gradual deterioration that would not be detected by the standard maintenance systems. The data is used to train clustering models that identify deterioration patterns and detect faults before the deterioration patterns result in failure or breakdowns.**

***Index Terms*—Internet of Things, Predictive Maintenance, Clustering, Dynamic Time Warping (DTW)**

I. Introduction

The widespread incorporation of Internet of Things (IoT) devices in various systems such as mobile phones, vehicles, and security systems to gather data has led to Industry 4.0 [1]. These large volumes of data, known as big data, prove to be very useful in gathering valuable insights about the user's experience and the device or component's performance. One such artificial intelligence application is called predictive maintenance, which uses such data has gained popularity and is being used by various organizations. Predictive maintenance uses IoT devices embedded in different kinds of devices and transport systems – road vehicles, railways, and aeroplanes – to collect data about their performance, and uses algorithms and methods such as active databases (rule-based), mathematical models, machine learning, and deep learning to predict values such as the remaining useful life (RUL) [2] of the machine or determine if the machine requires maintenance or repairs, that is, it predicts the deterioration of a machine or a group of machines that form a system. Prior to such a predictive approach, organizations would have to conduct scheduled check-ups for their vehicles to find out if the vehicles needed any repairs or maintenance; however, the scheduled check-ups resulted in downtime and shut-downs and so this proved to be costly and time-inefficient, as many of the vehicles would not require any kind of maintenance or repairs. Furthermore, unexpected breakdowns of vehicles resulted in the organizations incurring heavy costs. Such unexpected breakdowns are common when organizations do not conduct periodic checks to save resources and time. There are several maintenance strategies employed by organizations such as the aforementioned ones - they can be summarized into three primary categories – (i) corrective maintenance: this is a reactive measure in which a component or system has suffered failure and needs to be urgently repaired or replaced. Corrective maintenance occurs as a result of unexpected and unscheduled events; (ii) preventive maintenance: this is a measure that most organizations have in place to maintain the health and ensure the performance of the machinery. This measure involves conducting periodic checks to make sure that nothing is wrong with the equipment, and if a fault is found, maintenance activities are conducted accordingly; and (iii) predictive maintenance: this is a measure that estimates and predicts the probability of a component or equipment failing in the future. Such an approach allows organizations to avoid unexpected failures and skip non-required periodic check-ups that could be very costly.

In this paper, we provide an approach to conduct predictive maintenance by identifying the gradual deterioration (or gradual faults) of a public bus. Gradual faults are often not noticeable by the sensors on a vehicle nor visible to the eye until the deterioration progresses to a severe level, resulting in either component failure or immediate need for maintenance. The data used in this paper was collected in [3] from a public bus of the Société de Transport de l'Outaouais (STO) transit service of the Ottawa-Gatineau region. The data - sensor readings relevant to the cooling system of a bus - from a single bus are used to generate synthetic data to simulate 40 buses using Gretel Synthetic's [5] PyTorch implementation of the doppleGANger [4] (DGAN) model and then manually introducing temporal patterns and trends that were not captured by the generative model. After generating synthetic data, gradual deterioration in components is introduced to twenty percent of the buses. The faults introduced are based on common knowledge and heuristics. Finally, the generated data is used

Ontario Research Funds Grant Number: ORF-RE10-045

to train clustering models to identify the sensor readings with deterioration or faults. This way, individual models that identify abnormal sensor readings from the same bus can be grouped to identify the potential fault. The main contributions of this paper are as follows:

- A clustering approach to conduct predictive maintenance for a vehicle. The presented algorithm is tested on the cooling system of a bus, but it could be extended to several other machines and vehicle subsystems.
- A dataset containing synthetic sensor readings to simulate 40 buses along with additional test cases. Some of these buses have abnormalities (gradual faults) introduced. The dataset can be used to test machine learning algorithms - supervised and unsupervised for similar tasks.

This paper is structured as follows: Section 2 discusses work related to this study, Section 3 presents the proposed predictive maintenance algorithm, Section 4 describes the data used in this paper, Section 5 describes the experiments conducted using the proposed algorithm and the results, and Section 6 concludes this paper and points to some future directions of research.

II. Related Works

The concept of predictive maintenance has been around for a long time, however, the recent surge in available data and machine learning algorithms has increased its popularity. Several approaches are used to conduct predictive maintenance across various modes of transport such as mathematical models, rule-based approaches, machine learning, deep learning, or a combination of these. Many of these methods could be applied across different modes of transport if similar data is available. The work in [7] uses a rule-based approach to carry out the task of IoT-based predictive maintenance, and the works in [3], [8] use a rule-based approach for the task of deviation detection in a fleet - either on real data or on simulations. Methods that use the rule-based approach are not suitable for detecting gradual faults, as noise in the data could easily cross the threshold but that does not indicate a potential fault, additionally, the data values need to cross the threshold to determine a fault as opposed to the trends in the data being captured to observe slow degradation of components. The authors in [3] use the mathematical model - ICOSMO - for the task of sensor selection to be later on used as part of the algorithm. The work in [9] uses Multi-Target Probability Estimation algorithms to predict the probability and time of failure with the objective of notifying the user a week or two prior to any failure. Mathematical approaches such as maintenance scheduling, reliability estimation, and data selection are useful when included as part of the predictive maintenance algorithm for secondary tasks; relying on mathematical models solely for the task of degradation detection would not be feasible due to their sensitivity to noise and the range of values it was modeled for. The availability of real data allows for deep learning methods to be employed. However, deep learning approaches require large volumes of labeled data that are not easily accessible by the public; hence, relying on synthetic data becomes inevitable as well as the need to use traditional machine learning algorithms. The work in [10] used geographic data along with historic maintenance records to train a deep neural network along with autoencoders for the task of time-between-failures (TBF) prediction, which is the time for a machine between two consecutive breakdowns. Lastly, the authors in [11] use real sensor data to train supervised machine learning models such as decision trees, random forest, support vector machines, and neural networks to predict the clogging status of the oxygen sensor in an engine. The data used in this approach was collected in a controlled environment by exercising an engine and was then labeled by domain experts. Such a process is often not feasible for most organizations. Similar approaches have been used to conduct predictive maintenance in railways and aircraft. The popular components in railways are the tracks, the wheel bearings, and the pantograph, as these are considered to be the most prominent causes of failures in railways [12]. Approaches relying on historic maintenance data can employ similar machine learning models regardless of the mode of transport as long as there is sufficient data to train the model. The work in [13] proposes an architecture that uses clustering, tree-based algorithms, recurrent neural networks, and reinforcement learning to a) predict rail and geometry defects and b) determine required inspections. The authors used three datasets containing defects, inspection information, and load information. The work in [14] proposes an analysis system that uses clustering, an unsupervised learning technique, to perform predictive maintenance in aircraft, specifically in aero-engines. The approach is similar to the one presented in this paper, however, [14] primarily follows the idea of anomaly detection by finding points that are away from the dense region, as opposed to detecting specific potential faults in individual components; some aero-engines would have to be on the verge of failure for them to be marked as an anomaly in a large fleet of aircraft. Furthermore, they only use Euclidean distance as the metric for clustering, which may not give accurate results for data with noise and would be unable to capture and interpret complex trends.

III. Predictive Maintenance Algorithm

This section presents an algorithm to conduct predictive maintenance by using unsupervised clustering. K-Means clustering is used for detecting deterioration and unusual trends in the generated data. The metrics used to perform the clustering are Euclidean distance and Dynamic Time Warping (DTW) [15]. The primary idea is to form a) clusters containing normal buses, and b) clusters that identify component deterioration and faults. K-Means clustering using Euclidean distance is used as the baseline to compare to K-Means clustering using DTW. Euclidean distance is chosen as the base approach, because it is the common choice for clustering due to its low complexity and simplicity; it involves calculating the distance between a time series and the cluster centroid point-by-point without taking into account any trends or patterns. However, complex trends such as those found in coolant temperature

are difficult to distinguish using just a simple metric, which is why DTW is used. DTW temporally aligns sequences and calculates the distance between points that are similar even if the indices of the two points do not align in the time series. However, the complexity of DTW is high compared to the Euclidean distance method - $O(n*m)$, where *n* is the length of the first sequence and *m* is the length of the second sequence. Therefore, DTW should be used only if Euclidean distance fails to form accurate clusters. Additionally, the silhouette score is used to evaluate the quality of clusters, and the sum of squared distances is used to select the optimal value of K (number of clusters). The silhouette score is calculated as $b-a/max(a,b)$, where *a* is the mean distance of a point with the other points of its cluster and *b* is the distance between the single point and its nearest cluster. The sum of squared distances is the sum of the distances of each point in a cluster to the cluster centroid. Algorithm 1 provides an idea of how predictive maintenance can be conducted for a machine. The algorithm can be described as follows:

1) Select sensors relevant to a component or component group with the help of domain experts or knowledge base.
2) Restructure the sensor data so that each sensor is represented by a time series dataset containing several sequences, that is, a single time series dataset will belong to one sensor and will contain several instances or sequences, and each instance or sequence will contain multiple sensor readings over time.
3) Apply preprocessing steps such as reshaping, scaling, and cleaning the sensor data so that it can be used to train a model.
4) Choose a value of K (number of clusters) based on two methods – a) existing knowledge of the number of different types of faults to be detected or b) using the elbow method in which the sum of squared distances for various values of K are plotted and the value of K after which no significant decrease is observed is chosen.
5) Train two K-Means clustering models for each time series dataset - one using Euclidean distance as a metric and the other using DTW.
6) Use the silhouette score as a metric to evaluate clustering quality. If the silhouette score of the Euclidean model is greater than or equal to the score of the DTW model, select the Euclidean model as the final clustering model for that sensor, otherwise select the DTW model as the final clustering model for that sensor.
7) Retrieve the buses belonging to the cluster with faults from the chosen model by sorting the clusters by size and selecting any clusters after the largest one (the largest cluster will have all the normal sensor readings). Use the retrieved buses to identify the causes and types of faults with the help of domain experts to perform the necessary maintenance.

Algorithm 1 Predictive Maintenance Algorithm

Require: A dataset containing sensor readings
Ensure: Relevant Sensor Selection
$TimeSeries \leftarrow SensorData$ ▷ Restructure and reshape
Preprocessing($TimeSeries$)
if Types of faults known **then**
$K \leftarrow NumberOfTypes$
else
$K \leftarrow ElbowMethodOptimal$
end if
$EuclideanKMeans$.fit($TimeSeries$)
$DTWKMeans$.fit($TimeSeries$)
if $EuclideanSilScore \geq DTWSilScore$ **then**
$FinalModel \leftarrow EuclideanKMEans$
else
$FinalModel \leftarrow DTWKMEans$
end if
$FinalModelClusters$.sort(by=$size$, ascending=$False$)
$Faults \leftarrow FinalModelClusters[1:]$

IV. Data

The data used in this paper was first collected in [3]. The data consists of sensor readings from a bus of the STO transit service of the Ottawa-Gatineau region. The sensor readings from the bus were received as packets via the Society of Automotive Engineers (SAE) J1939 protocol [16], which were then decoded. The sensors chosen for this paper belong to the cooling system of the bus. The messages were transmitted and recorded at an interval of 1 millisecond to 100 milliseconds approximately and stored in a file of size 650 megabytes; the data was collected from a single bus over a duration of approximately ten hours in November 2018. Every J1939 packet contains data values from multiple sensors. Upon locating the relevant bytes for a sensor in the packet, the values need to be converted from hexadecimal to decimal and scaled. The J1939 specification document, which is published by the SAE, contains all the information that is required to retrieve and decode the physical data values of sensors. The specific sensors selected are Fan 2 Speed with rotations-per-minute units, Engine Coolant Temperature with degree Celsius units, and Engine Coolant Level 1 with percentage units. Due to sensor noise and availability issues, we did not include any additional cooling system-related sensors.

A. Data Processing and Generation

Three sensor readings were chosen for this work - Fan 2 Speed, Engine Coolant Temperature, and Engine Coolant Level 1 – as these were the only data bytes that contained values that seemed realistic and accurate. Upon decoding the values, the resultant dataset was stored in a comma-separated values (CSV) format, where each row had a timestamp associated with a J1939 packet. In order to generate synthetic data using a generative model with a Long Short-Term Memory (LSTM) network as the generator, categorical features had

to be removed and the dataset needed to be restructured and reshaped. The dataset was reshaped to be of the format [Fan Speed, Coolant Level, Coolant Temperature] as columns for the three sensors chosen. Reshaping the dataset resulted in only one sensor reading being in one row of the dataset (e.g., 4096.0, NaN, NaN), this is because decoded values were appended to the dataset row-by-row to keep the reshaped dataset in line with the time column. The null values were filled using the forward fill method in which an initial value is used to replace the following null values in a column until a non-null value is reached. All these steps resulted in a final time series dataset that contained three sensors, one in each column. The setup of the original dataset containing physical values from three sensors was ready after the reshaping; this dataset was used to generate synthetic data to simulate 40 buses and for the same up-time duration as the real data, which is for ten hours. The first step in the generation of synthetic data included using the DGAN model from Gretel [4]. The DGAN model is based on a Generative Adversarial Network (GAN) that uses an LSTM network as a generator. A few things to note about the model is that it does not support categorical features as mentioned previously so the dataset had to be restructured so that each column denotes a sensor and contains continuous values (decoded sensor readings). Secondly, the model only allows for a fixed length of sequences, and variable-length sequences were not supported. An example to explain the sequence length would be to consider a dataset that contains the temperature readings of a room every hour for seven days; such a dataset could contain a sequence length of twenty-four resulting in seven sequences of daily temperature readings. The data available for use in this work was only sensor data for a single bus so splitting the dataset into a fixed sequence length required repeated trying and testing of different values. Several combinations of hyperparameters were tested by trial and error to generate realistic data that also captured the temporal relations as well as included some randomness to simulate the stochastic aspects that lead to certain trends in the sensor readings. The hyperparameters configured were - the number of layers in the generator and discriminator, the number of units in an LSTM cell, the sequence length, the batch size, and the number of epochs. Finally, it was found that smaller batches and sequence lengths along with three or four layers in the network were the best combination. However, the resultant synthetic data failed to capture almost all of the temporal relations. This was evident in the sensor readings for the synthetic coolant temperature that were almost flat and constantly around 90 degrees Celsius in comparison to the real data, which contained temperature dips. Another instance of this was visible in the fan speed readings: the real data contained only two unique values - 4096 rpm and 0 rpm - with only fifty readings that had a value of 0 displaying that the fan was turned off for a few seconds across the whole time and the fan's operational value was at 4096 rpm. The synthetic data had no values of 0 rpm but consisted of values around 4096 rpm, which does not seem to be accurate for the functionality of an electric fan. One benefit of synthetic data generation was observed in the values of coolant level. The real sensor readings consisted of two values: 50% and 0; this indicates that the accurate value of the coolant level was 50% of the total container and there were a lot of readings in which the sensor was not able to record the real value so it was recorded as 0%. On the other hand, the synthetic data only had the actual value that was 50% but for all the generated buses, which does not hold true in real life, that is, buses could operate with varying levels of coolant without it causing a problem or failure. The main reason behind using the generative model to generate synthetic data was to get as close to the distributions and trends as possible. Although many temporal relations have not been captured, the generated data provides some useful insights such as a) there was no relation found between the fan turning off and the coolant temperature and levels, b) the coolant temperature's normal working temperature is around 90 degrees Celsius and the temperature dips needed an explanation, because the bus was running for almost the whole duration, and c) the coolant level values included 0% but this should be disregarded and filtered out. The aforementioned insights will be explained in the next section, because they are relevant to simulating the trends that were not captured by the generative model.

B. Manual Trend Introduction

In addition to the uncaptured trends, the synthetic data generated using the DGAN model contained a lot of fluctuations from one reading to the next, which made the synthetic data appear very noisy. The problem of noise was tackled in the data preprocessing step. To make the synthetic data more realistic, the temporal trends mentioned earlier that were not captured by the generative model needed to be manually introduced. An initial step made sure that all the data generated was within the permissible data ranges of the particular sensors, that is, the synthetic sensor readings could not have a value that is impossible to record, so all the data were fit between the minimum and maximum values allowed for each sensor. All generated buses had a coolant level of 50%, which is not true in real life, therefore, the coolant levels of the buses were changed based on probability - there was a 30% chance that the buses' coolant level would be increased by a value between 5% and 15% (resulting in the final values being between 55% and 65%) and a 15% chance that the coolant level would be increased by a value between 15% and 30% so the resulting coolant levels would be between 65% and 80%.

The coolant temperature dips that existed in the original data were missing in the synthetic data. The steep temperature dips in the real data from 90 degrees Celsius to approximately 60 degrees Celsius were justified by a few reasons: the engine speed was observed for the same timestamps as the temperature dips and a rotations-per-minute of zero during those timestamps suggested that the engine was not running whenever there was a large temperature dip. The sensor readings were recorded in the month of November when it is usually cold in Canada. The dip in temperature followed by

a rise took place over thirty minutes - these reasons imply that the bus was not running (turned off) for a period of approximately twenty minutes which resulted in the engine cooling down faster due to the cold weather. The introduction of temperature dips was based on the assumptions that a) a bus that was thought to be in perfect condition would have one break causing a significant temperature dip, b) a significant dip could be caused only in the cold months and not during summer, because it would take longer than twenty minutes for the engine to cool down and consequently the coolant, c) buses would not be turned off immediately after turning them back on (no consecutive temperature dips), and d) a bus would not have a temperature dip in the last twenty percent of its up-time due to a break. The dip sizes were either small (indicating that the bus was stationary for a small period of time in the cold), medium (indicating that the bus halted for a few minutes or was turned off for approximately five minutes, or large (indicating that the bus was turned off for a break). The sizes of the dips were determined by the number of time steps for which the temperature will keep decreasing and an appropriate number of time steps for it to come back to the functioning temperature. Furthermore, the temperature decline was not consistent or uniform; there was a very small probability that the temperature would go up by one degree, a higher probability that it would go down by one degree, and a significantly high probability that it would remain the same; this resulted in an inconsistent drop, which is how sensor recordings appear in real data. However, the temperature was not allowed to go below or above a set threshold based on acceptable coolant temperature values. Similarly, the temperature increase was not uniform and the probabilities of the temperature increasing, decreasing, or remaining the same were distributed as mentioned earlier but over fewer time steps.

The rotations-per-minute (rpm) at which the radiator fan operated was found to be 4096 rpm. Firstly, all the values that were not 0 rpm were made equal to the acceptable operation value - 4096 rpm. Secondly, as mentioned earlier, the generative model failed to capture the trend of the fan turning off (the rotations-per-minute being zero). An effort was made to find out the threshold behind the fan turning off by comparing the coolant temperature and fan speed at the same time steps, but there was no relation found between the coolant temperature decreasing and the fan turning off. The fan speed was also compared to the engine speed data, but the recorded speed data consisted of just noise, so no relation could be established apart from the readings with a value of 0 rpm, which implied that the engine was turned off. Therefore, the fan speed was set to zero for a few readings during the temperature drop, which seemed to be the most justifiable approach instead of randomly introducing fan turn-offs across the data.

The real data and the generated data with the trends introduced do not appear to look the same, but it still seems like a step in the correct direction to generate good-quality simulations as opposed to previous works in which synthetic data was generated by simply fitting the real data to a distribution and sampling from it such as in [3]. Moreover, all the simulated buses have unique trends depending on their coolant levels, temperature dip sizes and frequencies, and fan speed patterns. All the trends were introduced based on probability so there are also some instances in which there are no temperature drops, which implies that the bus was operating in summer and was running for the whole duration, consequently, the fan was not turned off for that bus up-time indicating that the fan ran throughout the duration of the bus' journey. The real bus data had only fifty recordings for which the fan was turned off, so the fan turning off was not a very important trend. The trends introduced cover many possibilities and scenarios, so if the synthetic data is used in the training of a model due to the unavailability of real data, the synthetic data will enable the model to be robust against seasonal and operational factors (that are acceptable and normal) without those factors being interpreted as anomalies.

C. Gradual Fault Introduction

After the trends were introduced to the generated synthetic data, the sensor data of 40 buses was simulated. However, the synthetic data so far was assumed to be of normal buses, that is, without any faults. Therefore, the next step was to introduce faults to the existing synthetic data. The faults introduced were gradual and subtle, in essence, they were not abrupt or large enough to trigger caution warnings, failures, or breakdowns in buses because those faults can be easily identified using a threshold-based approach. The assumption for fault introduction was that 20% of the buses in the dataset had potential faults; this assumption was made for simplicity to avoid imbalance problems due to the lack of large volumes of data required to tackle such a problem. Hence, eight buses had potential faults in them. The faults introduced were based on heuristics, common sense, and knowledge base. There were four types of faults introduced: the first fault involves lowering the coolant level of two buses by 10% and 15%, respectively, to indicate a potential leakage in the cooling system. The reduction of the coolant level was linear and over the duration of the buses' up-time, indicating that the coolant was leaking throughout the buses' trips. Note that the reduction was based on 15% of the initial value and not 15% of the entire coolant capacity. The second fault involves lowering the rpm at which point the fan of two buses operated by 30% and 35%, respectively, and increasing the coolant temperature slightly (by 5% and 7%, respectively) to indicate a fan performance deterioration. The decrease in the rpm was uniform and for all the values that were not 0%. On the other hand, the increase in the coolant temperature was linear and over the duration of the buses' just like the coolant level decrease. The third fault involves increasing the fan rpm of two buses by 10% and 15%, respectively, to indicate that the fan is rotating above the rated rpm of 4096 due to faulty wiring, a firmware glitch, or a similar issue, and it could cause wear-and-tear faster and lead to failure. Lastly, the fourth fault involves increasing the coolant temperature by 10% and 15%, respectively, to indicate a cooling system issue that cannot be caused by the coolant

level or fan performance. The purpose of this fault is to detect the temperature change that could have been caused by other cooling components for which the data could not be used in this work.

V. Cooling System Experiments and Results

The reason behind choosing a clustering approach is mainly the lack of real data with labels. Deep learning approaches have been rather popular amongst other methods due to the increasing computational powers of systems and the possibility of performing heavy computations at the cloud level instead of individual computers embedded in each vehicle. However, in cases of lack of labeled data, traditional machine learning algorithms perform incredibly well. Deep learning methods largely rely on the quality of data and its being labeled, furthermore, there is usually an imbalance in such tasks because most of the vehicles function normally and only a few have potential problems - it is difficult for deep learning methods to tackle such imbalance problems, especially without accurate and sufficient data. Therefore, choosing an unsupervised clustering approach favors efficiency and accuracy.

The clustering is performed for each sensor individually to identify any unusual patterns in the component that might not have any relation to the other components. The clustering results for each sensor for a bus can be combined to identify the type of fault and the cause of deterioration by experts. The preprocessing step includes converting all the data to integer type for memory efficiency, doing so results in no loss of information because the sensor recordings are whole numbers. The readings in which the coolant level value is 0% are removed, as they are noise. The coolant temperature data is noisy and has fluctuations so the moving average of the time series was calculated and used for smoothing the data. Additionally, only 10% of coolant level and fan speed readings for each bus were used by sampling at equal intervals; and even fewer readings of the coolant temperature to further make it less noisy and reduce memory usage. Using 10% of the dataset was sufficient as there were approximately 92,500 sensor readings for each bus in the dataset. The data for each sensor is then combined and formatted into time series datasets containing 40 buses. The coolant level dataset contained various values that are not relevant to any faults, for example, one bus may have a coolant level of 85% and another with just 50%, but that does not mean that there is anything wrong with either of them. Therefore, the shape of the time series was relevant to the problem at hand and not the magnitude of the values. Hence, the coolant level data was scaled to a distribution with zero mean and unit variance, doing so scaled all the normal data to have the same value (of zero) and trends could be observed between the values of 1 and -1. The choice of K (number of clusters) is based on the types of faults introduced for each sensor. The number of clusters for the coolant level is chosen to be 2, the number of clusters for the fan speed is chosen to be 3, and the number of clusters for the coolant temperature is chosen to be 2. The chosen values of K are also validated by using the Sum of squared distances for K values 2 to 5. The results show that the aforementioned chosen values of K based on the type of faults, are optimal, because the sum of squared distances does not decrease significantly when increasing the value of K beyond the optimal value. Moreover, the quality of the clusters is determined by using the silhouette score. The results of the Euclidean clustering and DTW clustering for the coolant level dataset are the same. They both perform incredibly well at identifying the decline in the coolant level so the Euclidean K-Means clustering model can be used for this problem for efficiency. The silhouette score of the K=2 cluster for the coolant level was 0.99, which is extremely accurate. Fig. 1 shows the two clusters, one contains all the normal buses and the other contains the two buses with a decline in the coolant level. The thick red line in all the following figures depicts the cluster centroid, while the less opaque gray lines depict individual time series.

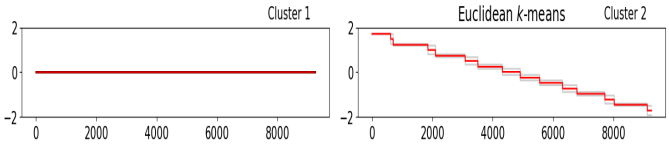

Fig. 1. K=2 Clusters for Coolant Level

Euclidean-based clustering fails to detect the slightly higher rpm in the fan speed dataset. The reason for this might be that the 0 rpm values in one bus might result in a large distance for congruent non-zero points in other buses (normal without faults), so the slight increase does not result in a large enough distance for it to be significant enough to be identified as a separate cluster. On the other hand, the DTW-based clustering was able to identify the two faults introduced to the fan speed and cluster them correctly, resulting in a silhouette score of 0.88. Fig. 2 shows the clusters using DTW and it is visible that all the faults are identified and clustered correctly.

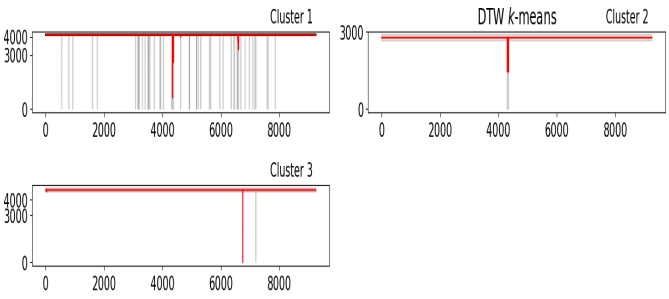

Fig. 2. K=3 Clusters (DTW) for Fan Speed

Clustering for the coolant temperature dataset was only performed using DTW. This is because the data has complex trends and noise that are not distributed uniformly. It is clear that using just the Euclidean method would not be enough to distinguish the trends from faults and give accurate results. The DTW clustering is able to identify the faults and cluster

them appropriately, although it fails to detect the rise in temperature in one bus. The rise in temperature is very subtle and is not detected by the model. Fig. 3 shows the clusters using DTW and it is visible that there is a time series in cluster 2 in which the temperature increases but it is clustered along with the normal buses. The silhouette score of the model is 0.64, which is reasonably high and is evident from the faulty buses being mostly clustered accurately.

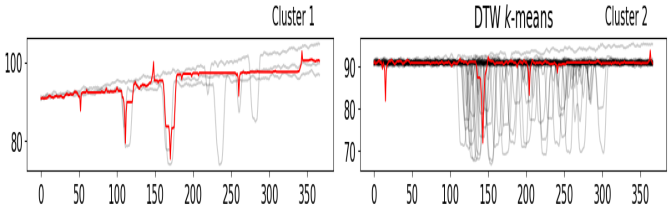

Fig. 3. K=2 Clusters (DTW) for Coolant Temperature

In summary, all the models perform as expected and are able to identify and cluster the faults appropriately. The model for the coolant temperature is unable to identify a single faulty trend but the model for the fan speed is able to identify the decreased rpm in the same bus that caused the increase in temperature, so when put together, all faults are accurately detected by the models.

VI. Conclusion

The work in this paper addresses the issue of common maintenance strategies being costly and inefficient. It shines a light on an alternative strategy that is cost and time-efficient - predictive maintenance - and presents an algorithm to perform it. The presented approach is tested out for the cooling system of a public bus. In this paper, we also generate a synthetic time series dataset to simulate a) buses that are normal, and b) buses with underlying faults and gradual deterioration that would not be identified by standard maintenance systems, because the faults have not crossed a threshold after which they result in component failure and damage. The aim of the dataset was to enable the training of a model that could detect these underlying faults accurately and distinguish them from the buses that are normal. Doing so could allow the user to take preemptive measures to have the problem rectified without resulting in any failures and breakdowns. Such an approach ensures efficiency in terms of cost, resources, and time, and most importantly the safety of the user. The clustering models in this work were trained mainly using synthetic data and for only three sensors, because other sensor data from the cooling system could not be used due to noise. A direction forward would be to gather real data from different bus models and in different weather conditions over a period of time. Furthermore, any buses experiencing faults could be labeled and the previous trips of those buses could be analyzed and used in the training of the model to find and predict the trends that resulted in failure. The data and models in this paper belong to the cooling system, however, the approach presented in this paper could be extended to any subsystem of a bus such as the braking system or the engine performance system. Creating models for these subsystems and combining them altogether could be used to perform predictive maintenance for a whole bus so that no faults could go undetected.

References

[1] N. Davari, B. Veloso, G. de A. Costa, P. M. Pereira, R. P. Ribeiro, and J. Gama, "A Survey on Data-Driven Predictive Maintenance for the Railway Industry," Sensors, vol. 21, no. 17, p. 5739, Aug. 2021, doi: https://doi.org/10.3390/s21175739.

[2] X. Si, W. Wang, C.-H. Hu, and D.-H. Zhou, "Remaining useful life estimation – A review on the statistical data driven approaches," European Journal of Operational Research, vol. 213, no. 1, pp. 1–14, Aug. 2011, doi: https://doi.org/10.1016/j.ejor.2010.11.018.

[3] P. Killeen, I. Kiringa, and T. Yeap, "Unsupervised Dynamic Sensor Selection for IoT-based Predictive Maintenance of a Fleet of Public Transport Buses," ACM Transactions on Internet of Things, Apr. 2022, doi: https://doi.org/10.1145/3530991.

[4] Z. Lin, A. Jain, C. Wang, G. Fanti, and V. Sekar, "Using GANs for Sharing Networked Time Series Data," arXiv (Cornell University), Oct. 2020, doi: https://doi.org/10.1145/3419394.3423643.

[5] "Gretel DGAN - Gretel.ai," Gretel.ai, https://docs.gretel.ai/create-synthetic-data/models/synthetics/gretel-dgan (accessed Feb. 21, 2024).

[6] J. Xie, J. Huang, C. Zeng, S. Jiang, and N. Podlich, "Systematic Literature Review on Data-Driven Models for Predictive Maintenance of Railway Track: Implications in Geotechnical Engineering," Geosciences, vol. 10, no. 11, p. 425, Oct. 2020, doi: https://doi.org/10.3390/geosciences10110425.

[7] R. Dhall and V. K. Solanki, "An IoT Based Predictive Connected Car Maintenance Approach," International Journal of Interactive Multimedia and Artificial Intelligence, vol. 4, no. 3, p. 16, 2017, doi: https://doi.org/10.9781/ijimai.2017.433.

[8] S. Byttner, T. Rögnvaldsson, and M. Svensson, "Consensus self-organized models for fault detection (COSMO)," Engineering Applications of Artificial Intelligence, vol. 24, no. 5, pp. 833–839, Aug. 2011, doi: https://doi.org/10.1016/j.engappai.2011.03.002.

[9] A. Sinaiski and H. S. Subramania, "Predictive Maintenance with Multi-target Classification Models," Lecture Notes in Computer Science, pp. 368–377, Jan. 2010, doi: https://doi.org/10.1007/978-3-642-12101-2_38.

[10] C. Chen, Y. Liu, X. Sun, C. D. Cairano-Gilfedder, and S. Titmus, "Automobile Maintenance Prediction Using Deep Learning with GIS Data," Procedia CIRP, vol. 81, pp. 447–452, 2019, doi: https://doi.org/10.1016/j.procir.2019.03.077.

[11] D. Giordano et al., "Data-driven strategies for predictive maintenance: Lesson learned from an automotive use case," Computers in Industry, vol. 134, p. 103554, Jan. 2022, doi: https://doi.org/10.1016/j.compind.2021.103554.

[12] M. Karakose and O. Yaman, "Complex Fuzzy System Based Predictive Maintenance Approach in Railways," IEEE Transactions on Industrial Informatics, vol. 16, no. 9, pp. 6023–6032, Sep. 2020, doi: https://doi.org/10.1109/tii.2020.2973231.

[13] P. C. Lopes Gerum, A. Altay, and M. Baykal-Gürsoy, "Data-driven predictive maintenance scheduling policies for railways," Transportation Research Part C: Emerging Technologies, vol. 107, pp. 137–154, Oct. 2019, doi: https://doi.org/10.1016/j.trc.2019.07.020.

[14] Y. Bao, G. Rui, and S. Zhang, "A Unsupervised Learning System of Aeroengine Predictive Maintenance Based on Cluster Analysis," Proceedings of the 2020 International Conference on Aviation Safety and Information Technology, Oct. 2020, doi: https://doi.org/10.1145/3434581.3434619.

[15] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (AAAIWS'94), AAAI Press, pp. 359–370, 1994.

[16] "SAE J1939 Standards Collection - SAE J1939 Standards Collection on the Web," SAE, https://www.sae.org/standards/development/ground-vehicle/sae-j1939-standards-collection-on-the-web (accessed Feb. 21, 2024).